```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv('/content/Mall_Customers.csv')
df.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

```
df.shape
```

```
(200, 5)
```

```
df.describe()
```

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

```
df.dtypes
```

```
CustomerID                 int64
Gender                    object
Age                        int64
Annual Income (k$)         int64
```

```
     Spending Score (1-100)     int64
     dtype: object
```

```
df.isnull().sum()
```

```
     CustomerID                0
     Gender                    0
     Age                       0
     Annual Income (k$)        0
     Spending Score (1-100)    0
     dtype: int64
```

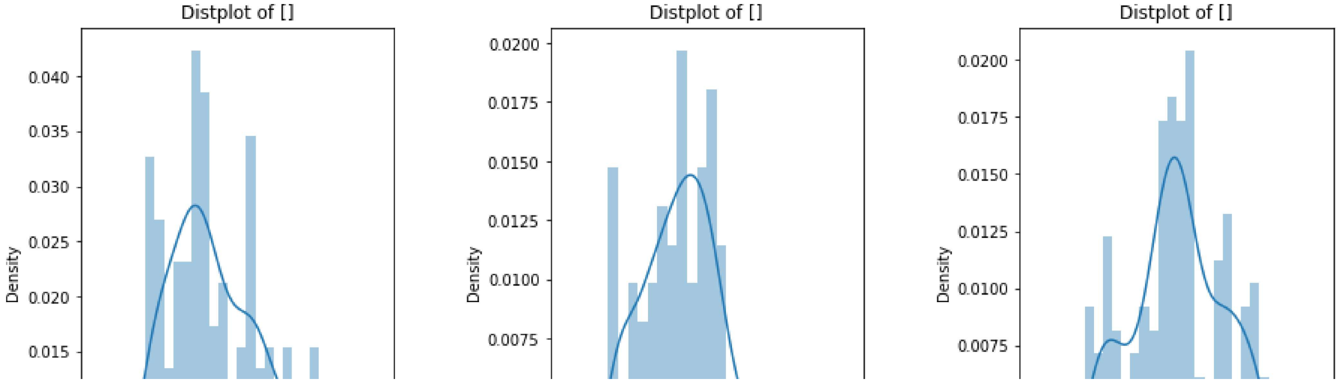
```
df.drop(["CustomerID"],axis=1,inplace=True)
```

```
df.head()
```

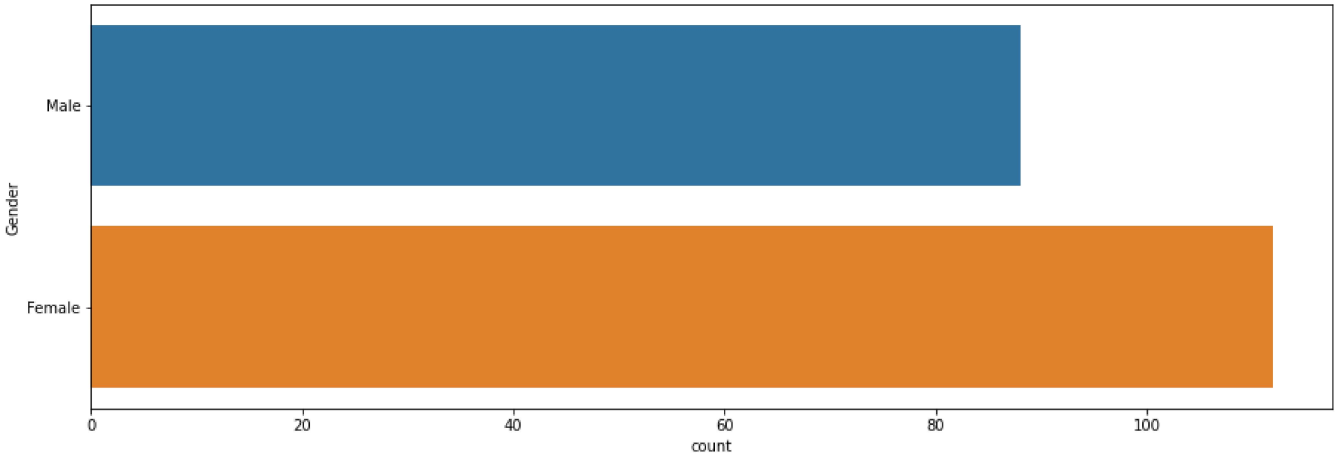
	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	Male	19	15	39
1	Male	21	15	81
2	Female	20	16	6
3	Female	23	16	77
4	Female	31	17	40

```
plt.figure(1, figsize=(15,6))
n=0
for x in ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']:
  n+=1
  plt.subplot(1,3,n)
  plt.subplots_adjust(hspace=0.5 ,wspace=0.5)
  sns.distplot(df[x], bins =20)
  plt.title("Distplot of []".format(x))
plt.show()
```

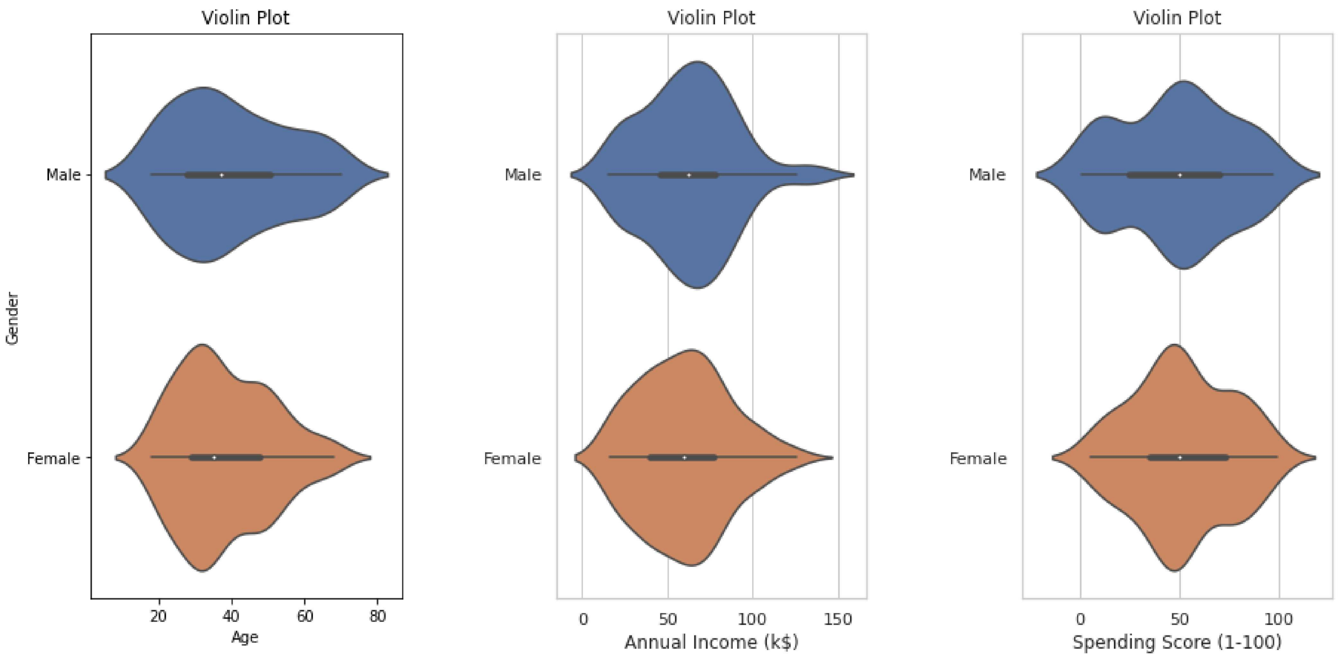
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarning:
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarning:
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2557: FutureWarning:
  warnings.warn(msg, FutureWarning)
```



```
plt.figure(figsize=(15,5))
sns.countplot(y='Gender',data=df)
plt.show()
```

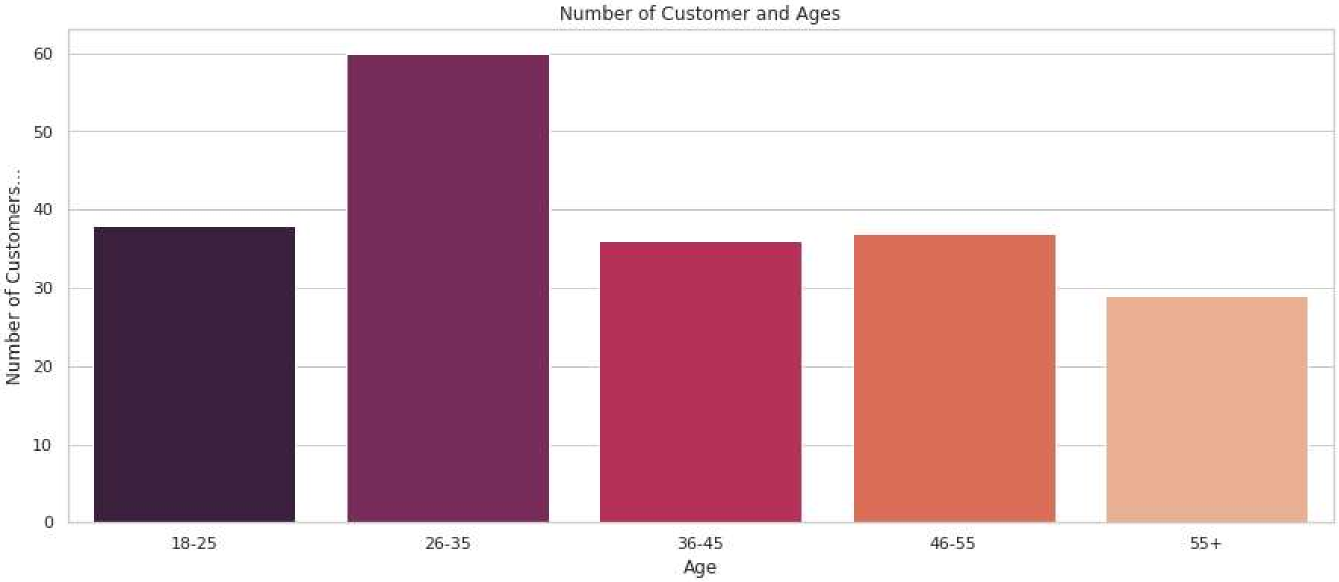


```
plt.figure(1, figsize=(15,7))
n=0
for cl in ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']:
  n+=1
  plt.subplot(1,3,n)
  sns.set(style='whitegrid')
  plt.subplots_adjust(hspace=0.5 ,wspace=0.5)
  sns.violinplot(x=cl,y='Gender',data=df)
  plt.ylabel('Gender' if n==1 else '')
  plt.title('Violin Plot')
plt.show()
```



Finding the range of age having highest number of customers.....

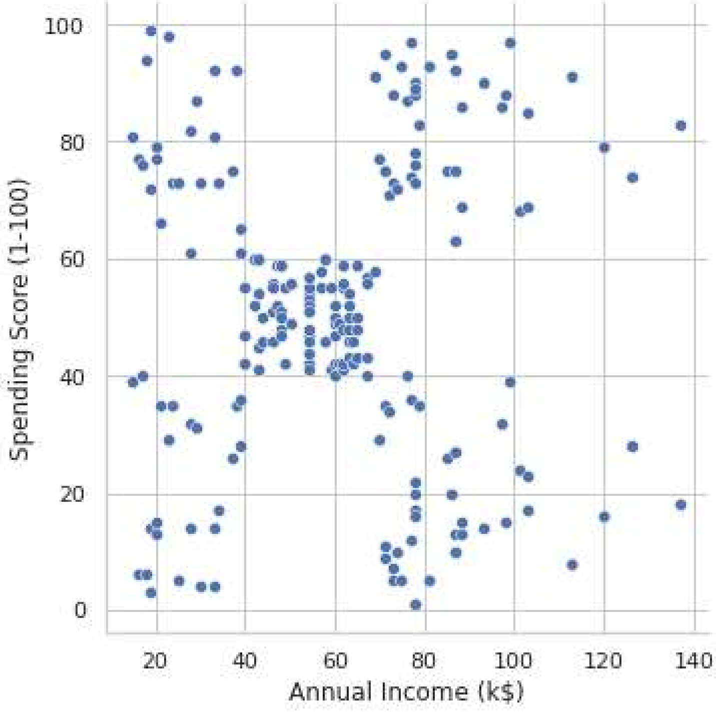
```
age_18_25 =df.Age[(df.Age>=18) & (df.Age<=25)]
age_26_35 =df.Age[(df.Age>=26) & (df.Age<=35)]
age_36_45 =df.Age[(df.Age>=36) & (df.Age<=45)]
age_46_55 =df.Age[(df.Age>=46) & (df.Age<=55)]
age_55above=df.Age[(df.Age>=56)]

agex=["18-25","26-35","36-45","46-55","55+"]
agey=[len(age_18_25.values),len(age_26_35.values),len(age_36_45.values),len(age_46_55.valu

plt.figure(figsize=(15,6))
sns.barplot(x=agex, y=agey, palette="rocket")
plt.title("Number of Customer and Ages")
plt.xlabel("Age")
plt.ylabel("Number of Customers...")
plt.show()
```



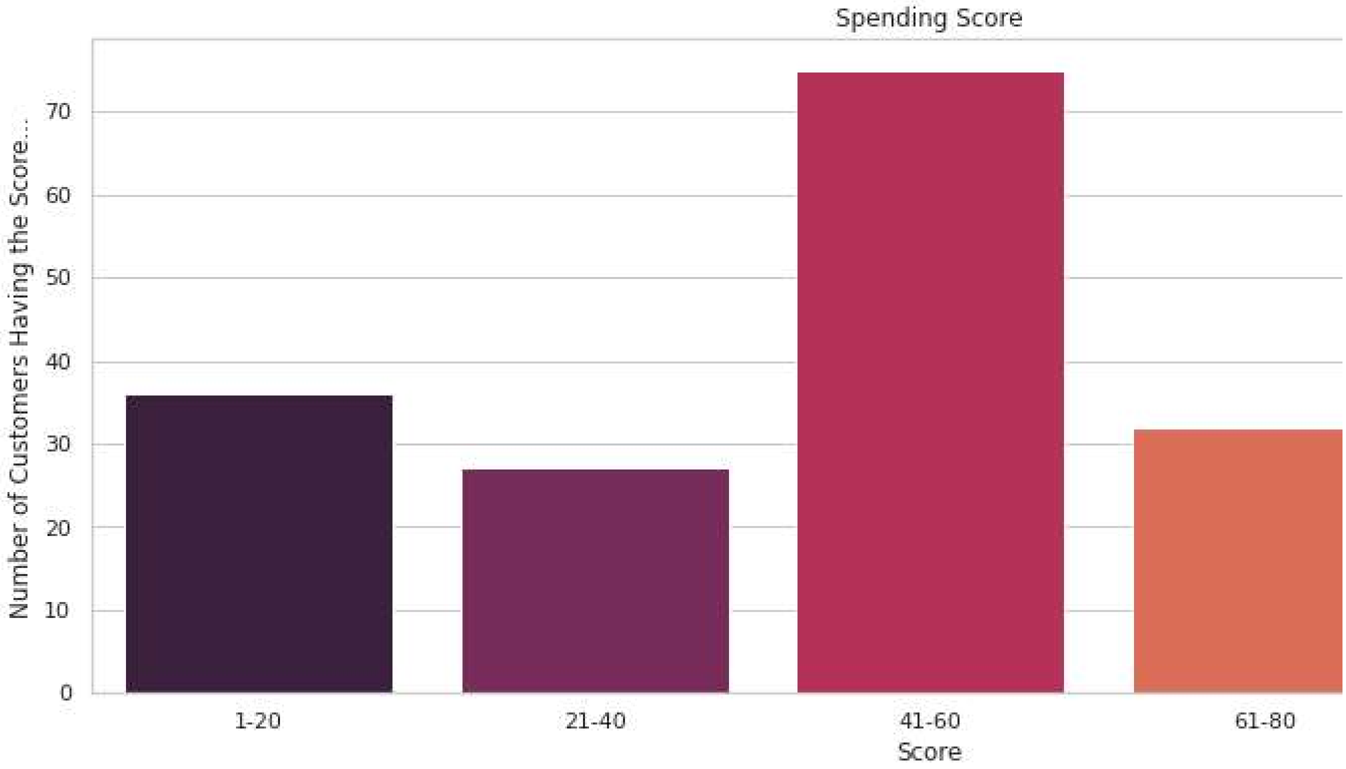
Finding the relationship between Annual income and Spending score....

```
sns.relplot(x="Annual Income (k$)", y="Spending Score (1-100)",data=df )
```

```
<seaborn.axisgrid.FacetGrid at 0x7ff62c236090>
```



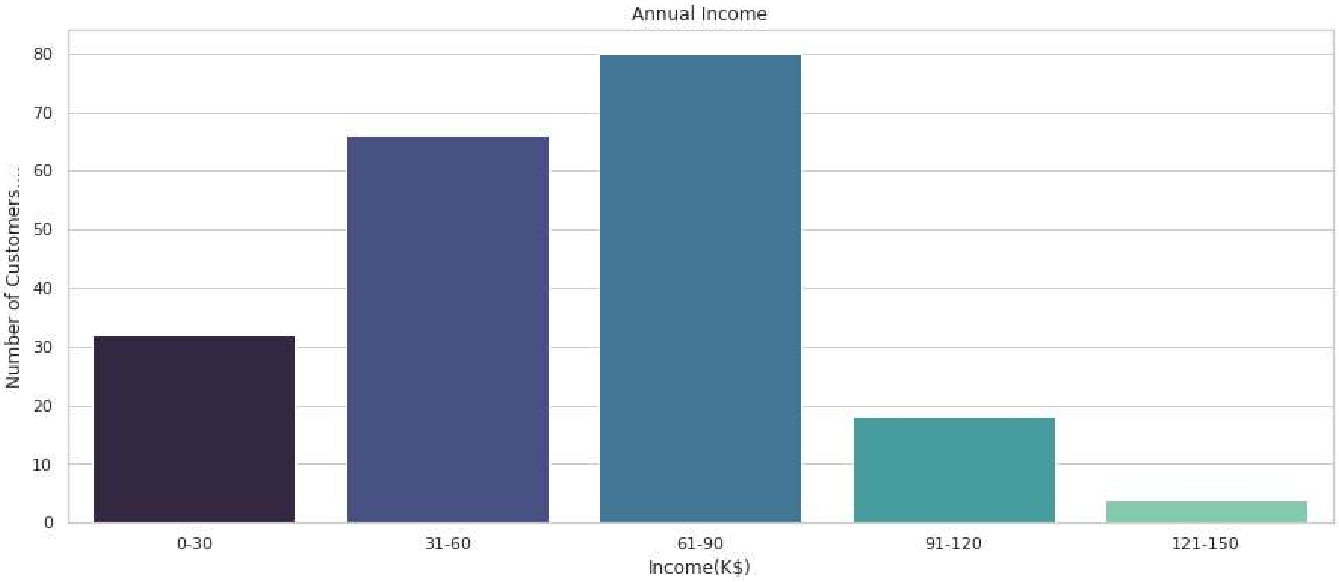
```
ss_1_20=df['Spending Score (1-100)'][(df["Spending Score (1-100)"]>=1) & (df["Spending Scc
ss_21_40=df['Spending Score (1-100)'][(df["Spending Score (1-100)"]>=21) & (df["Spending S
ss_41_60=df['Spending Score (1-100)'][(df["Spending Score (1-100)"]>=41) & (df["Spending S
ss_61_80=df['Spending Score (1-100)'][(df["Spending Score (1-100)"]>=61) & (df["Spending S
ss_81_100=df['Spending Score (1-100)'][(df["Spending Score (1-100)"]>=81) & (df["Spending 

ssx=["1-20","21-40","41-60","61-80","81-100"]
ssy=[len(ss_1_20.values),len(ss_21_40.values),len(ss_41_60.values),len(ss_61_80.values),le

plt.figure(figsize=(15,6))
sns.barplot(x=ssx, y=ssy, palette="rocket")
plt.title("Spending Score")
plt.xlabel("Score")
plt.ylabel("Number of Customers Having the Score...")
plt.show()
```



Repeating the above steps for Annual income.....

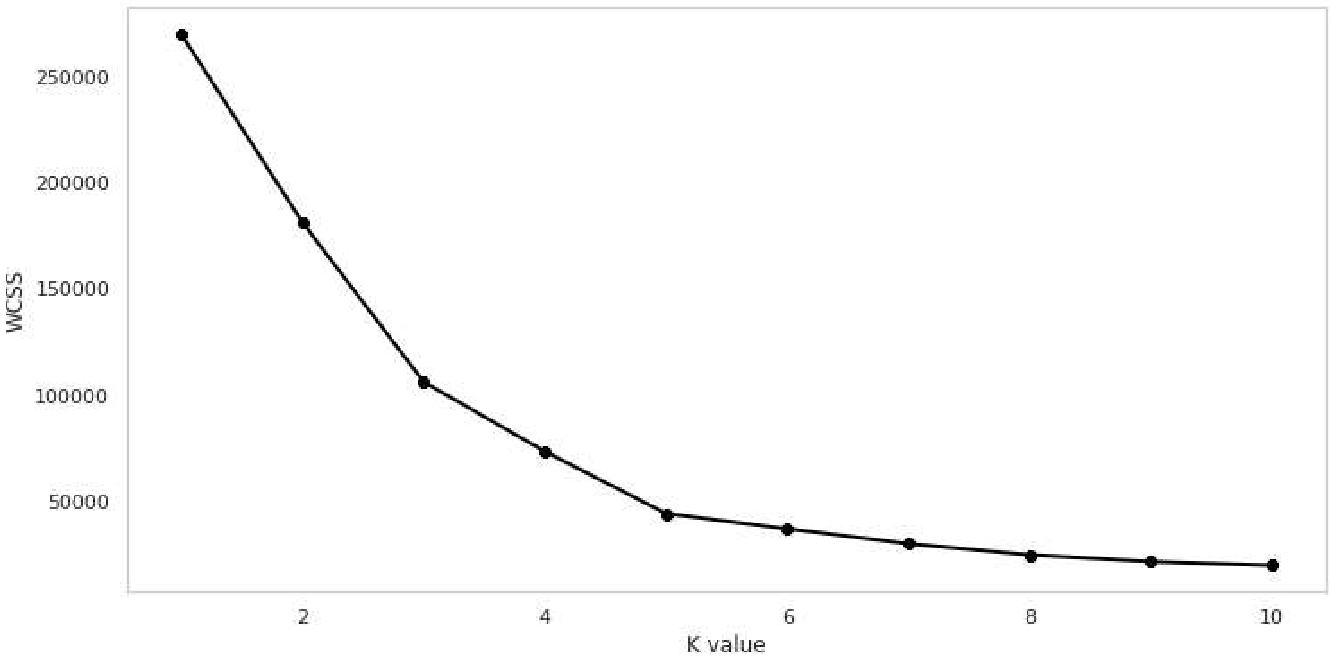
```
ai_0_30=df['Annual Income (k$)'][(df["Annual Income (k$)"]>=0) & (df["Annual Income (k$)"]
ai_31_60=df['Annual Income (k$)'][(df["Annual Income (k$)"]>=31) & (df["Annual Income (k$)
ai_61_90=df['Annual Income (k$)'][(df["Annual Income (k$)"]>=61) & (df["Annual Income (k$)
ai_91_120=df['Annual Income (k$)'][(df["Annual Income (k$)"]>=91) & (df["Annual Income (k$
ai_121_150=df['Annual Income (k$)'][(df["Annual Income (k$)"]>=121) & (df["Annual Income (

aix=["0-30","31-60","61-90","91-120","121-150"]
aiy=[len(ai_0_30.values),len(ai_31_60.values),len(ai_61_90.values),len(ai_91_120.values),l

plt.figure(figsize=(15,6))
```

```
sns.barplot(x=aix, y=aiy, palette="mako")
plt.title("Annual Income")
plt.xlabel("Income(K$)")
plt.ylabel("Number of Customers....")
plt.show()
```



Finding out the number of clusters we need to create.......

```
x1=df.loc[:,["Annual Income (k$)","Spending Score (1-100)"]].values

from sklearn.cluster import KMeans
wcss=[]

for k in range(1,11):
  kmeans = KMeans(n_clusters=k,init="k-means++")
  kmeans.fit(x1)
  wcss.append(kmeans.inertia_)

plt.figure(figsize=(12,6))
plt.grid()
plt.plot(range(1,11),wcss,linewidth=2,color="black", marker="8")
plt.xlabel("K value")
plt.ylabel("WCSS")
plt.show()
```



From the above Diagram it is clear that the values after 4th iteration remains constant....Hence ,we are roundoff the remaining clusters to be 4

```
kmeans =KMeans(n_clusters=5)

label =kmeans.fit_predict(x1)

print(label)
```

```
[4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4
 3 4 3 4 3 4 1 4 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 2 0 2 1 2 0 2 0 2 1 2 0 2 0 2 0 2 0 2 1 2 0 2 0 2
 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2 0
 2 0 2 0 2 0 2 0 2 0 2 0 2 0 2]
```

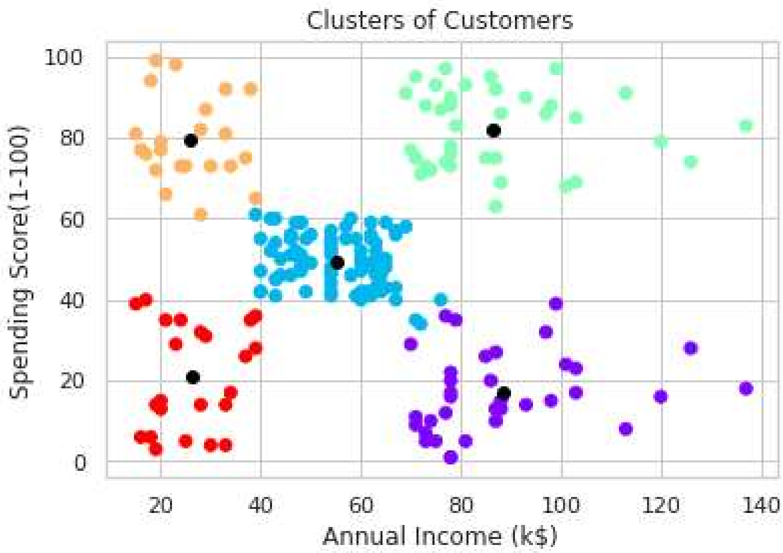
We can see that that is confined to four values i.e, 0,1,2 and 3

```
#The centroids are:::

print(kmeans.cluster_centers_)
```

```
[[88.2        17.11428571]
 [55.2962963  49.51851852]
 [86.53846154 82.12820513]
```

```
     [25.72727273 79.36363636]
     [26.30434783 20.91304348]]
```

```
plt.scatter(x1[:,0],x1[:,1], c=kmeans.labels_, cmap='rainbow')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],color ='black')
plt.title('Clusters of Customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score(1-100)')
plt.show()
```


Clusters of Customers
Spending Score(1-100)
Annual Income (k$)